Jonathan Derin (s0554306)
Marcel Grewe (s0553951)
Georg Donner (s0553821)
Christian Westphal (s0554200)

HTW Berlin / Datenbanken
WS 16/17

Erarbeitung einer Datenbanklösung für einen Fußballscout

Betreuer	Pro. Dr. Rüger Oßwald
Ort, Datum	Berlin, 11.01.2017

1. Kurzbeschreibung

Für einen Fußballscout soll eine Datenbank mit verschiedenen Spielern der verschiedenen deutschen Vereine, ausschließlich aus der 1. Bundesliga, erstellt werden. Hierbei besteht die Anforderung, dass der Scout alle transferrelevanten Informationen der Spieler auslesen kann. Er soll zudem einen Überblick über alle getätigten Transfers bekommen können. Außerdem sollte der Scout die Möglichkeit haben, Statistiken wie zum Beispiel die Spielerbewertung der Spieler in der Liga auswerten zu können.

2. Vorraussetzungen

2.1. Technik & Software

Das Projekt wurde erstellt mit Hilfe von:

- PowerDesigner - für die Modellierung der konzeptuellen und physischen Ebene
- SQLAnywhere - für die Implementierung der Datenbank

2.2. Implementierungshinweise

Die Datenbank enthält nur einen Ausschnitt der Vereine, Trainer, Spieler und Transfers der 1. Fußball Bundesliga. Die Trainer der jeweiligen Vereine bleiben dem Verein treu und wechseln zu keiner Zeit. Die Spielerbewertung wird nicht berechnet sondern jährlich durch Experten vergeben. Der Scout tritt über eine Internetrecherche an den Verein heran, daher haben wir auf die Kontaktinformationen der Spieler und des Vereins bewusst verzichtet.

3. Funktionsumfang

Selectabfragen:

- Welche 10 Spieler beziehen das meiste Geld?
- Welche 5 Spieler haben die längste Zeit nicht gewechselt?
- Welche Vereine haben weniger als 10 Spieler?
- Welche Torhüter haben eine Bewertung über 80?
- Wie hoch sind die Gesamtausgaben (Gehalt) eines Vereins pro Jahr?
- Wie hoch sind die Gesamtausgaben der Vereine durch Transfers?
- Wer ist der Spieler mit den häufigsten Transfers?
- Wie hoch ist die durchschnittliche Spielerbewertung?

View:

- zeige alle aktiven Spieler an

Funktionen:

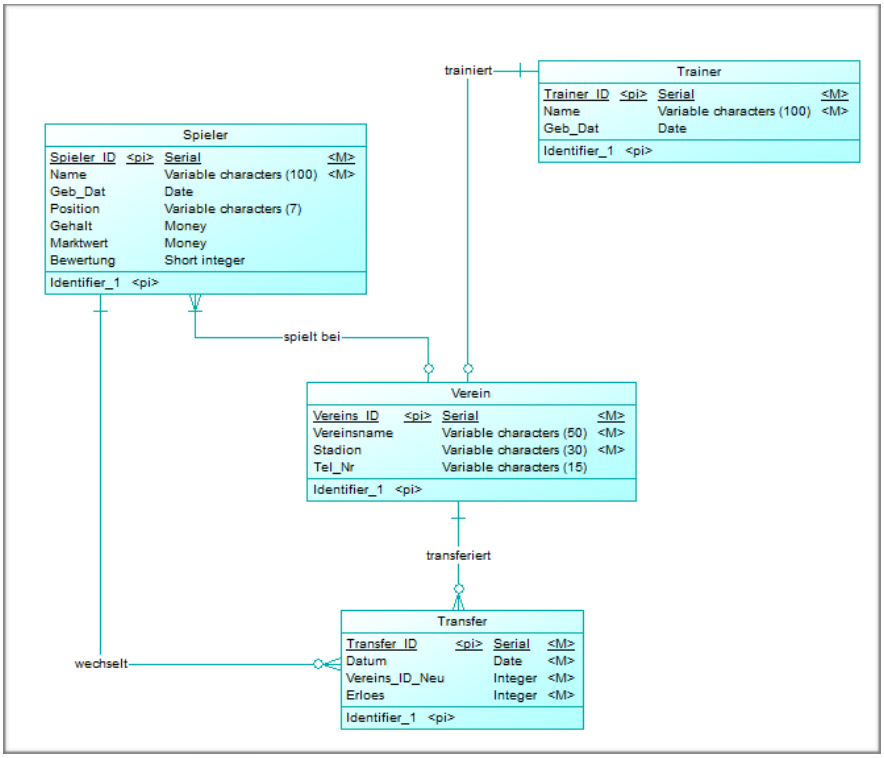
- Zeige die meisten Transfers
- Wer ist der beste Spieler?

Prozeduren:

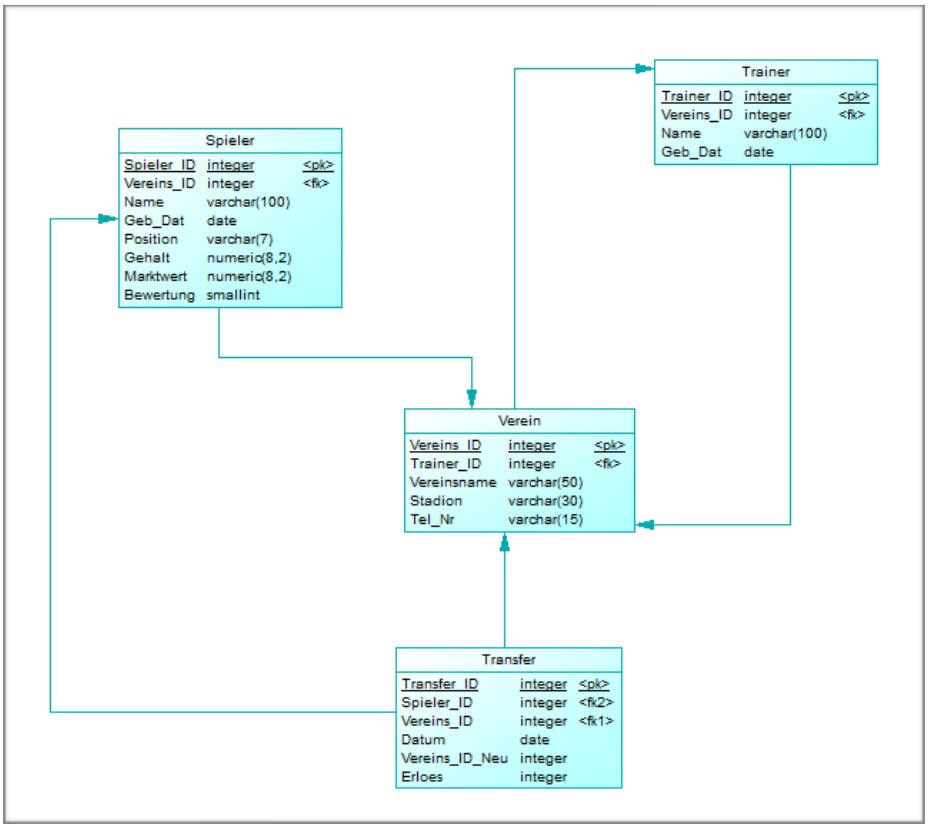
- neuen Spieler anlegen
- Spielerdaten ändern
- Spieler transferieren
- Spieler in den Ruhestand schicken

4. Datenbankentwurf

4.1. Konzeptuelle-Ebene



4.2. Physische-Ebene



5. View

```
ALTER VIEW "DBA"."AktiveSpieler"
as
(select *
from Spieler
where VEREINS_ID is not null
order by vereins_ID)
```

Die View “AktiveSpieler” zeigt alle Spieler, die aktuell einem Verein zugeordnet sind. Spieler ohne einer Vereins_ID werden dabei nicht berücksichtigt.

6. Funktionen

6.1. “meisteTransfers” - Funktion

```
ALTER FUNCTION "DBA"."meisteTransfers" ()
 returns integer
  as begin
    declare @tmpSumme integer = 0
    declare @summe integer = 0
    declare @vereinID integer = 0
    declare @len integer = 0
    declare @counter integer = 0
    SELECT @len = count(*) from transfer

    while @counter<=@len
     BEGIN
        SELECT @tmpSumme = count(TRANSFER.VEREINS_ID_NEU) from transfer where TRANSFER.VEREINS_ID_NEU = @counter
            if @tmpSumme>@summe
             begin
               set @summe = @tmpSumme
               set @vereinID = @counter
             end


      set @counter = @counter+1
     END

    declare @vname varchar(50)
    declare @ges integer = 0
    SELECT @vname = VEREIN.VEREINSNAME from verein where VEREIN.VEREINS_ID = @vereinID
    SELECT @ges = sum (TRANSFER.ERLOES) from TRANSFER where TRANSFER.VEREINS_ID_NEU = @vereinID group by VEREINS_ID_NEU

    message 'Der Club mit den meisten Transfers (' ||@summe|| 'Stk.) ist '  ||@vname|| ' Gesamterlös: ' ||@ges|| '€' to client
    return @summe
  end
```

Hier wird ermittelt, welcher Verein die meisten Transfers mit dem dazugehörigen Gesamterlös durchgeführt hat.

Parameter: @tmpSumme integer, @summe integer, @vereinID integer, @len integer, @counter integer, @ges integer, @vname varchar,
Returns: Anzahl maximaler Transfers
Ausgabe: Clubname, Anzahl Transfers, Gesamterlös
>> execute meisteTransfers

6.2. “bestspieler” - Funktion

```
ALTER FUNCTION "DBA"."bestspieler" (@pos varchar(7))
 returns integer
   as begin
    declare @sp_id integer
    declare @spname varchar (100)
    SELECT top 1 @sp_id = SPIELER.SPIELER_ID from spieler where Spieler.Position = @pos order by BEWERTUNG desc
    Select @spname= SPIELER.NAME from SPIELER where SPIELER.SPIELER_ID = @sp_id
    message 'Bester ' || @Pos || ' = '  ||@spname   to client
  return @sp_id
  end
```

Hier wird der beste Spieler anhand seiner Spielerposition und -bewertung ermittelt.

Parameter: @Pos varchar, @sp_id integer, @spname varchar,
Returns: Spieler_ID integer
Ausgabe: Spielername, Spielerposition
>> execute bestspieler 'ST'

7. Prozeduren

7.1. “SpielerErst” - Prozedur

```
ALTER PROCEDURE "DBA"."SpielerErst" @VereinsID integer,@SPname varchar(100),
@gebDat date,@Pos varchar(7),@gehalt money,@mktwrt money,@bew smallint
as begin
insert into Spieler
values (default,@VereinsID,@SPname,@gebDat,@Pos,@gehalt,@mktwrt,@bew)
end
```

Alle Attribute, außer SpielerID, werden automatisch als Parameter chronologisch gesetzt.

Parameter: @VereinsID integer, @SPname varchar, @gebDat date, @Pos varchar, @gehalt money, @mktwrt money, @bew smallint

>> execute SpielerErst 2, ‘Neymar', ‘1990/8/2', ‘LF', 1500, 15000, 92

7.2. “SpielerEdit” - Prozedur

```
ALTER PROCEDURE "DBA"."SpielerEdit" @SPname varchar(100),
@Pos varchar(7),@gehalt money,@mktwrt money,@bew smallint
as begin
update Spieler

    set POSITION = @POS,
        GEHALT = @gehalt,
        MARKTWERT = @mktwrt,
        BEWERTUNG = @bew
    where Name = @spname

end
```

“SpielerEdit” verändert die Daten der Spieler, da z.B. Gehalt kein statischer Wert ist.

Parameter: @SPname varchar, @Pos varchar, @gehalt money, @mktwrt money, @bew smallint

>> execute SpielerEdit ‘Neymar','ST',1550,12000,91

7.3. “SpielerRuhest” - Prozedur

```
ALTER PROCEDURE "DBA"."SpielerRuhest" @SPname varchar(100)
as begin
update Spieler

    set VEREINS_ID = null
    where name = @spname

end
```

Hier wird der Spieler nicht gelöscht, da die Daten in Transfer sonst unbrauchbar wären (beziehen sich auf den Spieler). Die Vereins_ID wird dabei auf *null* gesetzt.

Parameter: @SPname varchar

>> execute SpielerRuhest 'Andres Iniesta’

7.4. “SpielerTransfer” - Prozedur

```
ALTER PROCEDURE "DBA"."SpielerTransfer" (@spName varchar(100),
@vereinNeu varchar(50),@pos varchar(7),@gehalt money,@preis money)
 as begin
    declare @sp_ID integer =  0
    declare @vereinIDAlt integer = 0
    declare @vereinIDNeu integer = 0

    SELECT @sp_id = SPIELER.SPIELER_ID from spieler where SPIELER.NAME = @spName
    SELECT @vereinIDAlt = SPIELER.VEREINS_ID from spieler where SPIELER_ID = @sp_id
    SELECT @vereinIDNeu = VEREIN.VEREINS_ID from verein where VEREIN.VEREINSNAME = @vereinNeu

    insert into TRANSFER
    values (default,@sp_id,@vereinIDAlt,now(),@preis,@vereinIDNeu)

    update Spieler
    set SPIELER.VEREINS_ID = @vereinIDNeu ,
        SPIELER.POSITION = @pos,
        SPIELER.GEHALT = @gehalt
    where SPIELER.SPIELER_ID = @sp_id
 end
```

Die Prozedur “SpielerTransfer” ermöglicht den Transfer eines Spielers zu einem neuen Verein und wird in *Transfer* gespeichert. Somit entsteht eine Historie aller Transfers.

Parameter: @spName varchar, @vereinNeu varchar, @pos varchar, @gehalt money, @preis money

>> execute spielerTransfer 'Lionel Messi’, 'Borussia Dortmund’, ‘ST', 1000, 50000

8. Selectabfragen

Welche 10 Spieler beziehen das meiste Geld	SELECT TOP 5 * FROM spieler ORDER BY marktwert DESC
Welche 5 SPieler haben die längste Zeit nicht gewechselt	SELECT TOP 5 name, position, gehalt, marktwert, bewertung, datum, erloes FROM spieler, transfer WHERE spieler.spieler_ID = transfer.spieler_ID ORDER BY datum
Welche Vereine haben weniger als 10 Spieler	SELECT DISTINCT vereinsname FROM verein, spieler WHERE (SELECT COUNT(*) FROM verein, spieler WHERE spieler.vereins_ID = verein.vereins_ID) <10

Welche Torhüter haben eine Bewertung über 80	SELECT * FROM spieler where position = 'TW' AND bewertung > 80 ORDER BY bewertung DESC
Gesamtausgaben (Gehalt) eines Vereins pro Jahr	SELECT vereinsname, SUM(gehalt) FROM verein, spieler WHERE spieler.VEREINS_ID = verein.VEREINS_ID GROUP BY vereinsname ORDER BY SUM(gehalt) DESC
Gesamtausgaben der Vereine durch Transfers	SELECT vereinsname, SUM(erloes) FROM verein, transfer WHERE transfer.vereins_ID_neu = verein.VEREINS_ID GROUP BY vereinsname ORDER BY SUM(erloes) DESC
Spieler mit den häufigstens Transfers	SELECT FIRST name, COUNT(*) as transfers FROM spieler, transfer WHERE transfer.SPIELER_ID = spieler.spieler_ID GROUP BY name ORDER BY transfers
Durchschnittliche Spielerbewertung	SELECT vereinsname, AVG(bewertung) AS 'Durschnittl. Bewertung' FROM spieler, verein WHERE spieler.vereins_ID = verein.vereins_ID GROUP BY vereinsname

9. Verbesserungs- & Erweiterungsmöglichkeiten

Die von uns erstellte Datenbank bietet die grundlegenden Informationen, die ein Scout benötigt, um neue Spieler für einen Verein zu akquirieren. Die Trainer wurden für unser Datenbankmodell aussenvorgelassen, da es den Rahmen für dieses Projekt sprengen würde.
In der Realität wäre der Trainer unverzichtbar, da es selbstverständlich auch Trainerwechsel gibt. Die Datenbank könnte noch durch Hinzufügen weiterer deutscher als auch internationaler Ligen erweitert werden. Des Weiteren könnte man Spieltage mit den jeweiligen Platzierungen der Vereine, Heim- und Auswärtsspiele, sowie Niederlagen und Siege hinzufügen. Um den Spieler besser bewerten zu können, wären Informationen über gespielte Pässe, Ballbesitz, Tore, gewonnene Zweikämpfe, erhaltene gelbe als auch rote Karten mit der entsprechenden Spielsperrung und gehaltene Elfmeter (Torhüter) sinnvoll. Derzeit arbeiten wir nur mit der durch Experten vergebene Spielerbewertung. Am Ende könnte man noch sagen, wer der Torschütze oder Spieler des Jahres ist.

10. Quellen

- Skript der DatenbankenVorlesung
- Interactive SQL Help